RAIRO-Oper. Res. 54 (2020) 101–117
https://doi.org/10.1051/ro/2018069

**RAIRO Operations Research**
www.rairo-ro.org

# A RULE-BASED HEURISTIC ALGORITHM FOR ON-LINE ORDER BATCHING AND SCHEDULING IN AN ORDER PICKING WAREHOUSE WITH MULTIPLE PICKERS

Mehrdad Alipour[1], Yahia Zare Mehrjedrdi[1,*] and Ali Mostafaeipour[1]

**Abstract.** In manual order picking systems, orders received from internal or external customers are collected by pickers that walk or ride through the warehouse. Generally, orders are grouped into several sub sets, *i.e.* batches to reduce picking time and cost. This paper considers the on-line order batching problem (OOBP) in an order picking warehouse with multiple pickers in which the maximum completion time of all batches (customer orders) has to be minimized. At first, a mathematical model is introduced for the off-line version of the problem. Then the on-line version of the problem is considered in which customer orders become available dynamically over time. Since the proposed model is NP-hard, a rule-based heuristic algorithm was proposed to solve the on-line problem. The main contributions of the present work is to propose a mathematical model for the off-line order batching problem with multiple pickers considering makespan minimization and to present a novel heuristic algorithm for solving the on-line version of the problem. To validate the proposed algorithm, it is proved that its competitive rate is equal to 2. Finally, the solution algorithm is evaluated through a series of experiments and the most appropriate routing, batching and selection policies are introduced.

**Mathematics Subject Classification.** 90B06, 90B35.

Received November 13, 2017. Accepted September 4, 2018.

## 1. Introduction

Orders specified by internal or external customers pass through several physical processes in a warehouse before being shipped to customers which include receiving orders, putting away, order picking, checking and packaging. Order picking is the process of retrieval of items from their storage locations in a warehouse in order to satisfy customer demands. Since incoming items in a warehouse are received and stored in large-volume and customers order small volumes, order picking problem arises. The negative impact of unsatisfactory customer service as a result of long processing and delivery times on the one hand and high labor and delivery costs on the other hand, may reduce the competitiveness of the warehouse [11].

Order picking systems can be classified into two categories: picker-to-part systems and parts-to-picker systems. In the first type of order picking systems, automated storage and retrieval machines deliver the required items to

*Keywords.* Warehouse, order picking system, order batching, on-line order batching, picker-to-part systems.

[1] Department of Industrial Engineering, Yazd University, Yazd, Iran.
[*]Corresponding author: yzare@yazd.ac.ir; mehrjerdyazd@gmail.com

stationary pickers while in the second type of order picking systems, pickers walk or ride through the warehouse and collect the required items [17].

Considering the second type of order picking systems, three different planning problems arise at the operational level [2]: the assignment of items to storage locations (article location), grouping customer orders into picking batches (order batching) and routing of pickers through the warehouse (picker routing). This paper focuses on the second problem in which different customer orders can be combined into picking orders (batches) to increase the efficiency of warehouse operations considerably [4].

With respect to the available information from customer orders, two different kinds of order batching problems can be distinguished [20]: off-line (static) order batching problem in which all customer orders are known at the beginning of planning period and on-line order batching problem (OOBP), where customer orders are not known in advance and become available dynamically over time. In this paper, an on-line order batching problem with multiple pickers is considered in which three main decision issues need to be solved: (1) which customer orders should be grouped in the same batch; (2) how should the constructed batches be assigned to available pickers; and (3) when should the picking process of each batch be started. The objective is to minimize the maximum completion time of all batches (makespan). Since the off-line order batching problem is NP-hard [7], a rule-based heuristic approach is proposed to solve the aforementioned decision issues. The algorithm breaks down the on-line order batching problem into some off-line problems regarding the defined decision point types and solves each off-line order batching problem based on several defined decision rules which consist of batching rule, batch selection rule and picker assigning rule.

The remainder of this paper is organized as follows. In Section 2, we provide a review of the relevant literature. In Section 3, we establish a mathematical model for the off-line order batching problem with regard to multiple pickers. In Section 4, to solve the on-line version of the proposed model, a rule-based heuristic algorithm will be presented. To analyze the solution quality of the proposed algorithm, a competitive analysis of the generated solutions will be implemented in Section 5. Moreover several examples are tested to verify the effectiveness of the proposed model and algorithm and to evaluate the performance of the proposed on-line algorithm when different methods are used for order batching, batch selection and picker routing in different problem classes in Section 6. Conclusions and further research topics will be discussed in Section 7.

## 2. Literature survey

The global optimum for order picking problems with picking time minimization only can be achieved by considering three planning problems (article location, order batching and picker routing) simultaneously, but such an integrated model is not a realistic approach [17]. In practice, most of order batching methods consider picker's routing as constant.

Gademann and Velde [7] presented a branch-and-price algorithm to solve the small instances of the off-line order batching problem in reasonable computational time. A mixed integer programming approach is proposed by Bozer and Kile [1] to obtain near-optimal solutions for the off-line order batching problem. The proposed approach is only applicable for instances with small number of orders (up to 25). Heuristic approaches should be used to solve larger instances of off-line order batching problems in reasonable computational time.

There are four type of heuristic approaches in the literature that can be used to solve off-line order batching problems: priority rule based algorithms, seed algorithms, saving algorithms and meta-heuristic algorithms. In the first type of algorithms, priority rule based algorithms, a rank is assigned to customer orders based on their priorities and then orders are assigned to batches according to the defined ranks [8]. One method to assign priorities to orders is to utilize FCFS (First Come First Served) rule. In the second type of order batching algorithms, seed algorithms, an order is selected as a seed for each batch using a seed selection rule and then other orders are added to the seed order according to an order-congruency rule [6]. In this type of algorithms, batches are created sequentially. In the methods of the third group, saving algorithms, for each pair of customer orders, the saving that can be obtained by combining all items of the two customer orders instead of considering them as two separate batches are calculated. Then the pair of customer orders with maximum saving value are combined to shape a single batch and be picked in a larger picking tour. This approach is repeated until no further improvement can be obtained. The last group of algorithms are the meta-heuristic algorithms. Hsu *et al.* [12] proposed a genetic algorithm to solve the off-line Order Batching Problem but their approach is limited to S-Shape routing policy. Tsai *et al.* [15] presented a multiple genetic algorithm with earliness and tardiness penalties to solve the integrated order batching and picker routing problems and to obtain the best picking plan. Henn *et al.* [11] also proposed an algorithm which is a combination of an iterated local search and an ant colony algorithm. Henn and Schmid [10] showed that how meta-heuristics can be used to minimize the total tardiness for a given set of customer orders with predetermined due dates.

Considering on-line order batching problem, Kamin [13] considered the problem of retrieving greeting cards from a warehouse using automated guided vehicles. The author simulated and evaluated the system considering different objectives including the makespan in each day and also different batching strategies and problem parameters. In their work, the picking priorities of batches are defined according to the due dates (a point in time when the customer order is due to be completed) of customer orders they consist of.

Won and Olafsson [18] considered an order batching problem in which customer orders arrive at different times. The authors observed a trade-off between total service time and turnover time of an order. They proposed an optimization model where the objective function is the weighted sum of service time and waiting time of customer orders within a batch. To be able to use this approach, they supposed that arrival time of customer orders are known in advance. FCFS rule and 2-Opt procedure is applied to solve the order batching and picker routing problems, respectively.

Elsayed and Lee [5] described a warehouse with automated storage and retrieval system where customer orders become available dynamically over time and each order has a due date. They presented a model to construct batches from customer orders and to sequence picking of batches with regard to minimizing tardiness of customer orders. The authors considered problem in two cases: static and dynamic. In the static version of the problem, customer orders that arrive in a particular time interval constitute a new batch. This time interval is defined based on the time needed to process the previous batch. In the dynamic case, an order arriving while a batch is being processed results in rescheduling of storage and retrieval process. The proposed batching algorithm sequences customer orders with respect to their due dates and utilize a nearest schedule rule, a shortest service time rule, and a locations rule to select batches for picking.

Recently time window batching have been used frequently to model the on-line order batching problem. Chew and Tang [3] applied variable time window batching to model OOBP considering S-Shape routing policy. In variable time window batching, the order picker waits until a specified number of orders has received then collects them as a batch in a tour. The authors applied theoretical analysis and modeled the order picking system as a queuing network with two queues to estimate travel and service times. The first queue is to model batching process in which customer orders are sequenced according to FCFS rule and are serviced (batched) when a particular number of orders are in the queue. The second queue represents the picking process. The orders that have been batched are moved to the second queue and released sequentially according to availability of order pickers.

Le-Duc and de Koster [14] assumed uniform distribution for demand frequency and estimated the average turnover time for a random order among customer orders. The authors showed that the average turnover time of a random order is a convex function of the batch size. It means that on the one hand, when the batch size increases, average service time of each order decreases but average waiting time of each order increases and on the other hand, a small batch size leads to a large average service time, but to a small average waiting time.

Van Nieuwenhuyse and de Koster [16] proposed a queuing model to calculate average turnover time of an order in a 2-block warehouse considering fixed and variable time window batching.

Henn [9] proposed a MILP formulation to minimize the maximum completion time of customer orders for an OOBP in which customer orders arrive within a certain time period. According to the presented model, there is one order picker in the warehouse to pick customer orders using S-Shape and Largest Gap routing policies. The author modified previously proposed approaches for off-line order batching algorithms to deal with the on-line situation. They also conducted analytical analysis to find upper and lower bounds for competitive ratio of the proposed algorithm.

The OOBP which is considered by Henn [9] is similar to the on-line batch processing problem (OBPP) with the objective of minimizing the makespan. In OBPP jobs arrive dynamically over time and a machine can process them as a batch in a way that the starting times and the completion times of orders in the same batch are identical. In the capacitated OBPP each batch may contain a limited number of jobs. The difference between OOBP and OBPP is that the processing time of a batch in the OBPP is defined according to the longest processing time of any job in the batch while in the OOBP the processing time of a batch is defined according to all the orders assigned to the batch. The author proposed an approach based on the existent off-line algorithms for the OBPP to solve the OOBP.

Xu *et al.* [19] described a warehouse with pick-and-sort picking system considering random storage policy, S-Shape routing policy and FCFS rule as batching policy. The authors proposed an analytical model to approximate the average throughput time of an arbitrary customer order. They assumed that the average throughput time of a customer order consists of the average waiting time for the order to be batched (TW), the total service time for a batch in the picking area (TP) and the total service time for a batch in the sorting area (TS). It is proved that the average throughput time of an arbitrary order is a convex function of the batch size. Computer simulation is used to analyze the effect of different parameters on the expected customer throughput time and the batch size.

Zhang *et al.* [21] studied the integrated on-line order batching and distribution scheduling in an e-commerce environment. The authors assumed that the distribution operations are outsourced to a 3PL service provider and as such, fixed departure times are considered for vehicles. They proposed a heuristic approach to solve the on-line order batching problem with fixed vehicles departure times.

In this paper we extend the model proposed for OOBP by Henn [9] to consider multiple pickers (OOBPMP) since considering one picker limits the exploration of picking efficiency. A mathematical model will be presented for the off-line version of the problem to: (1) construct batches from customer orders; (2) assign constructed batches to order pickers; (3) define the start time of picking each batch, such that the maximum completion time of all batches is minimized.

## 3. Problem description and formulation

### 3.1. Problem description

The picking process in an order picking warehouse can be described briefly as follows: the order picker starts at the depot with a pick list which represents the storage locations of the requested articles and the number of items demanded for each article (a customer order or combination of customer orders). He/she walks or ride by a vehicle through the warehouse and collects items from different storage locations according to the pick list. Then he/she returns to the depot and delivers the picked items. The route that order picker walks or rides through it to pick the items is determined by a routing policy. There exists optimal polynomial time algorithm for picker routing in the literature [9] but, it has been rarely practical because order pickers refuse optimal routes due to their confusing nature. Among picker routing policies, S-Shape and Largest Gap heuristics which provide near-optimal straightforward routes have been used frequently in practice [9].

Since number of arriving orders is too large to process each order separately as a single batch in an appropriate time, customer orders are combined to form batches. Customer orders can be combined to form a batch until the capacity of picking device is not violated, since order pickers usually pick items with the help of a picking device. Order integrity condition has to be satisfied, *i.e.* splitting the orders among more than one batch is prohibited, since it would result in additional sorting effort. Interruption of order picking process to rearrange the batches according to newly arrived orders is not also allowed.

The time it takes to complete a batch is called batch service time. The service time of a customer order is defined according to the service time of the batch which the customer order is assigned to. The service time of a batch includes walking time, picking time, and setup time. Walking time is the time period it takes from

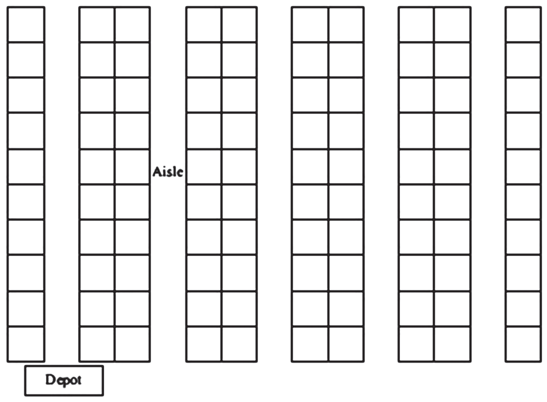

Figure 1. The layout of picking area.

order picker to travel from the depot to the first pick location, between the pick locations and from the last pick location to the depot. Pick time is the time period needed to put the items from the pick locations on the picking device. Setup time is the time period it takes to do setup-tasks at the first and the last pick locations in each tour.

In the order picking system which is considered in this paper, limited number of pickers are employed to retrieve customer orders during a shift. Customer orders are not known in advance during the shift and arrive dynamically over time. These decisions have to be made without information about upcoming orders: (1) which customer orders should be assigned to the same batch; (2) which picker should process each batch; and (3) when should picking process of each batch be started.

The point in time when a customer order becomes available and when a batch is assigned to an available order picker are called order arrival time and batch starting time, respectively. The starting time of an order is identical to the starting time of the batch which includes the order. The point in time when an order picker returns to the depot after collecting all the items of a batch is called the completion time of the batch. The completion time of an order is also equal to the completion time of the batch the order is assigned to. The waiting time of a batch or a customer order can be defined as the time period between customer order arrival time and starting time. The time period that a customer order stays in the picking system *i.e.* the time period between the customer order arrival time and completion time is also called the customer order turnover time.

In the following, we present a mathematical model for the off-line version of OOBPMP with the objective of minimizing the maximum completion time of all batches.

## 3.2. Optimization model

The mathematical model for the OOBPMP is based on the following assumptions:

- The layout of the picking area is a common single-block warehouse with two cross-aisles which is shown in Figure 1.
- S-Shape and Largest Gap routing policies are utilized to define picking routes of order pickers.
- Class-based storage policy is applied to store items in the warehouse. According to this storage policy, products are classified based on their demand frequency and each class of products is assigned to a dedicated area of warehouse. Storage allocation within an area is random.
- Customer orders are unknown in advance and become available dynamically over time. A rule based batching heuristic is proposed to batch arriving orders such that makespan is minimized. Each customer order must be incorporated into one batch.
- An order picker can handle a batch only when the previous batch which is assigned to him/her is finished.

- Multiple order pickers are considered to increase the efficiency of order picking process. It is assumed that all pickers are identical, *i.e.* Walking, searching and picking velocity for all order pickers are the same.
- Aisles in the warehouse are wide and pickers that have to pick an item from the same storage location, will not block each other.

In the following, an optimization problem is formulated for the off-line version of the OOBPMP which requires the complete information of all incoming orders. The off-line model is presented to analyze the structure of the optimization problem. The following constants are used in the model:

| | |
|---|---|
| $n$ | Number of customer orders |
| $m$ | Upper bound on the number of required batches (a trivial upper bound can be $m = n$) |
| $k$ | Number of order pickers |
| $r_i$ | Arrival time of customer order $i$ ($\forall i \in \{1,\dots,n\}$, where $0 \le r_i \le r_{i+1}$) |
| $t_{\text{setup}}$ | Setup time, time of set-up tasks for each batch |
| $v_{\text{travel}}$ | Travel velocity, number of length units the order picker can walk per time unit |
| $v_{\text{pick}}$ | Pick velocity, number of items the order picker can search and pick per time unit |
| $w_i$ | Number of items requested in customer order $i$ ($\forall i \in \{1,\dots,n\}$) |
| $W$ | Picking device capacity, which is the maximum number of items per batch |

The following variables are used in the model:

| | |
|---|---|
| $x_{ij}$ | 1 if customer order i is assigned to batch j, 0 otherwise |
| $s_{jl}$ | Start time of picking batch j by order picker l |
| $\text{st}_j$ | Service time of batch j including walking time, searching time and picking time |
| $\text{dis}_j$ | Distance function of picking tour for batch j, which is defined by routing policy |

The off-line version of the OOBPMP can be formulated as follows:

$$\text{minimize} \max_{j\in\{1,\dots,m\}} \left\{ \text{st}_j + \sum_{l=1}^{k} y_{jl} s_{jl} \right\} \tag{3.1}$$

$$\sum_{j=1}^{m} x_{ij} = 1, \qquad \forall i \in \{1,\dots,n\} \tag{3.2}$$

$$\sum_{i=1}^{n} x_{ij} \le n \sum_{i=1}^{n} x_{i(j-1)}, \qquad \forall i \in \{1,\dots,n\} \tag{3.3}$$

$$\sum_{l=1}^{k} y_{jl} = 1, \qquad \forall j \in \{1,\dots,m\} \tag{3.4}$$

$$\text{st}_j = t_{\text{setup}} + \frac{\text{dis}_j}{v_{\text{travel}}} + \frac{\sum_{i=1}^{n} w_i x_{ij}}{v_{\text{pick}}}, \qquad \forall j \in \{1,\dots,m\} | \sum_{i=1}^{n} x_{ij} \neq 1 \tag{3.5}$$

$$\sum_{i=1}^{n} w_i x_{ij} \le W, \qquad \forall j \in \{1,\dots,m\} \tag{3.6}$$

$$s_{jl} \ge \max_{i\in\{1,\dots,n\}} \{r_i x_{ij}\}, \qquad \forall j \in \{1,\dots,m\}, l \in \{1,\dots,k\} \tag{3.7}$$

$$s_{jl} \ge s_{(j-1)l} + y_{(j-1)l}\text{st}_{(j-1)l}, \qquad \forall j \in \{1,\dots,m\}, l \in \{1,\dots,k\} \tag{3.8}$$

$$s_{jl} \geq 0, \qquad \forall j \in \{1,\dots,m\}, l \in \{1,\dots,k\} \tag{3.9}$$

$$x_{ij}, y_{jl} \in \{0,1\}, \qquad \forall i \in \{1,\dots,n\}, j \in \{1,\dots,m\}, l \in \{1,\dots,k\} \tag{3.10}$$

The objective function (3.1) minimizes the maximum completion time of all batches or all customer orders. The completion time of picking a batch is obtained by adding the service time of the batch to the start time of the batch. Equation (3.2) ensures that each customer order is assigned to exactly one batch. Inequality (3.3) guaranties that batches are opened sequentially. Equation (3.4) ensures the assignment of each batch to exactly one order picker. Equation (3.5) calculates the service time for each batch in the order picking system which. The service time of a batch is composed of setup time, the time the order picker needs to travel through the warehouse and the time he/she needs to search for the items in the batch and pick them. Service time is calculated only for batches which at least one customer order is assigned to them. Inequality (3.6) guaranties that the capacity of the picking device is not violated. Inequality (3.7) indicates that an order picker can start the picking of a batch when all customer orders assigned to this batch are already known. Inequality (3.8) indicates that an order picker can start picking of a batch only when he/she has finished the picking of the previous batch which is assigned to him/her. Equations (3.9) and (3.10) represent that start times of batches are non-negative and that variables $x_{ij}$ and $y_{jl}$ are binary variables.

In the presented model, for an empty batch $j$, according to equation (3.7), we will have $s_{jl} \geq 0$ for all $l \in \{1,\dots,k\}$. Since the objective function in equation (3.1) is of minimization type, all $s_{jl}$ for the aforementioned empty batch $j$, are allowed to take the least possible value which is zero. Furthermore, there is no limit on the service time of an empty batch and it is also allowed to take the least possible value which is equal to zero. As a result, an empty batch will have no effect on the objective function. Moreover, we need to show that in equation (3.8) a batch which is not assigned to an order picker will not affect start time of batches which are assigned to that picker. Consider the first order picker for example. It is clear that for batch $j$ which is assigned to another picker, $y_{1j}$ will be zero and according to equation (3.8), inequality $s_{1j} \geq s_{1(j-1)}$ holds. Since the objective function is of minimization type, equality $s_{1j} = s_{1(j-1)}$ holds. Considering constraint (3.8) for batches $j-1$ and $j-2$ in the same way, one obtains $s_{1j} = s_{1(j-1)} \geq s_{1(j-2)} + y_{1(j-2)}\mathrm{st}_{1(j-2)}$. So, it has been shown that a batch which is not assigned to an order picker will not affect other batches which are assigned to him/her.

## 4. Rule-based heuristic algorithm

In this section we consider the on-line order batching problem with multiple pickers in which we have no information on the arrival times of upcoming orders. Since the off-line version of OOBPMP is NP-hard and there is a need to solve the on-line problem at some decision points in time in reasonable time, a novel rule-based heuristic algorithm is proposed to form batches, assign them to appropriate order pickers and schedule the batches which are assigned to each order picker in a way that the completion time of all batches (makespan) is minimized. We will define three types of decision points and explain the basic principles of solutions in Section 4.1. Batching, selection and assigning rules will be explained in Sections 4.2–4.4, respectively.

### 4.1. Basic principle

We distinguish three types of decision points in the order picking system as follows:

Type A: The point in time at which at least one order picker becomes idle whereas a set of unprocessed orders (also called open batches) are available.
Type B: The point in time at which an order arrives whereas at least one order picker is idle.
Type C: The point in time at which the last customer order arrives.

At each decision point, a solution to the off-line version of the problem needs to be obtained. Since the off-line order batching problem in NP-hard [9], a batching heuristic rule namely $H_B$ is employed to form batches. When the batches are generated and entered the batch operating system, a selection rule namely $H_S$ and three assigning rules $H_A^1$, $H_A^2$ and $H_A^3$ are employed to select and assign the generated batches to the order pickers. To define the picking route for each batch, S-Shape and Largest Gap heuristics are used which are the most common routing heuristics in the literature. The basic principle of the on-line algorithm is summarized in Algorithm 1.

**Algorithm 1**. Basic principle of the on-line order batching algorithm.

```
**At time t (each decision point Type A, B or C)**
Unprocessed orders are entered into batch generation system
**If** there is an unprocessed order which is not assigned to any batch **then**
A set of batches are generated using batching heuristic rule $H_B$
**End if**
Batches are entered into batch operation system
Batches are scheduled by means of assigning rules ($H_A^1$, $H_A^2$ and $H_A^3$) and
selection rule $H_S$
```

## 4.2. Batching rule ($H_B$)

For the batching heuristic rule ($H_B$) there are several options in the literature which can be used. Three heuristic algorithms namely the First-Come-First-Served rule (FCFS), the saving algorithm (C&W(ii)) [4] and iterated local search [11] which are the common batching algorithms in the literature are employed to generate batches from customer orders at each decision point.

According to the FCFS rule the customer orders are ranked based on their arrival times and are assigned to batches with respect to batch capacity constraint and their rank.

In the first step of C&W(ii) a different batch is constructed for each customer order. Then a saving value is calculated by combining each pair of batches. This saving value can be obtained in picking distance or time by picking the items of both batches in a common tour instead of picking them in different tours. The pair of batches with the largest saving value are selected and are combined if their combination will not violate the capacity constraint. If two batches were combined, the saving values is computed again for the pairs of batches. This process continues until when there is no possible combination of batches or there is no pair with positive saving value.

Iterated local search is a kind of local search in which the search process starts from a candidate solution and the algorithm tries to search iteratively for better neighborhood solution through two phases namely improvement phase and perturbation phase. Improvement phase starts from an initial solution and returns a local optimum. The algorithm then explores the vicinity of this local optimum to obtain an improved solution with better objective function.

## 4.3. Selection rule ($H_S$)

Four different selection rules are suggested in the literature namely FIRST, SHORT, LONG and SAV. According to the FIRST selection rule, the first constructed batch is selected to be assigned to one of the order pickers. For example if batching is implemented by using FCFS rule, a batch whose orders have the minimal arrival time will be selected first. According to the SHORT selection rule, a batch with the shortest service time will be selected and assigned to the order pickers first. Batches with short service time usually does not involve items that are stored in aisles far from the depot. When the demand frequency of items in the aisles far from the depot is low, it may be more efficient to pick them at the end of the planning period. LONG selection strategy selects the batch with the longest service time. Using this strategy will increase the time interval between the completion times of two consecutive batches. As a result, more customer orders may arrive during picking time of a batch which can be combined with the non-processed customer orders and shape more efficient batches. SAV strategy intends to combine customer orders with more similarity in their storage locations. In order to measure similarity between orders in a batch, SAV computes a saving value for each batch which is equal to the sum of the single service times of the assigned customer orders and subtracts the batch service time.

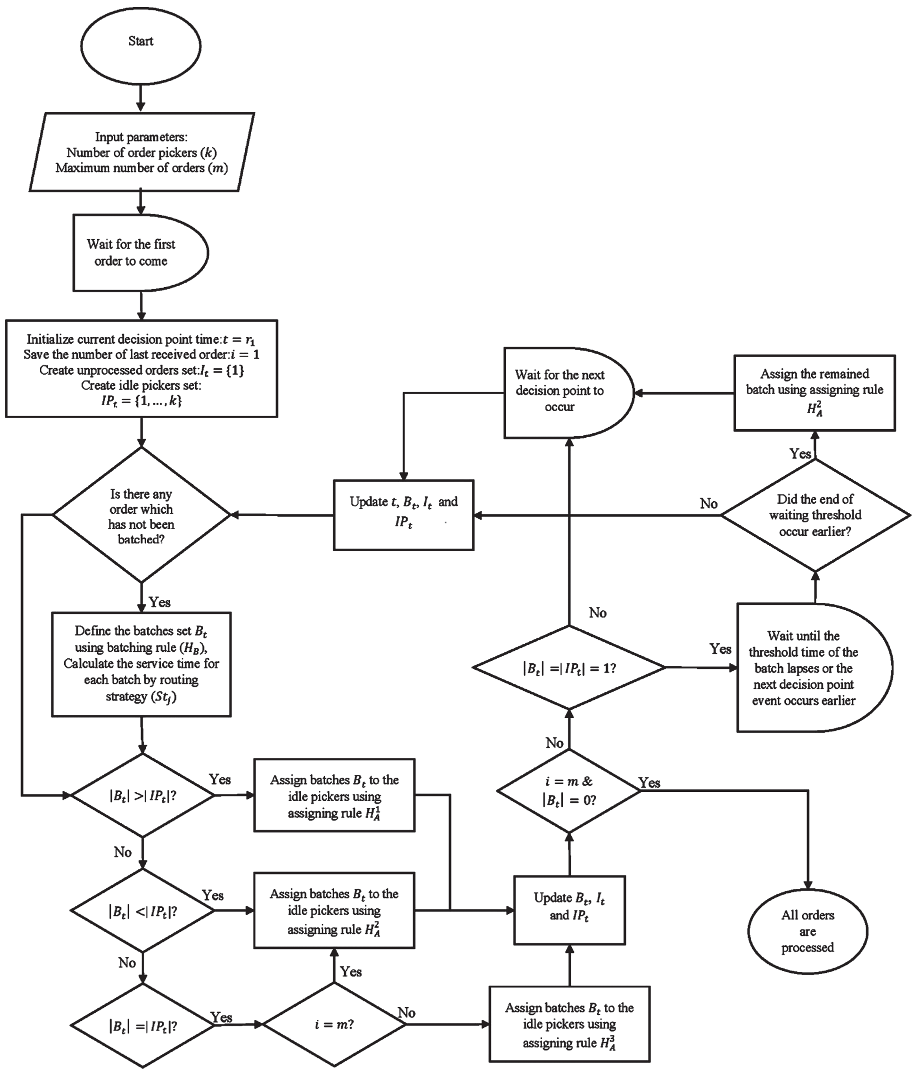
Start
Input parameters: Number of order pickers ($k$) Maximum number of orders ($m$)
Wait for the first order to come
Initialize current decision point time: $t = r_1$ Save the number of last received order: $i = 1$ Create unprocessed orders set: $I_t = \{1\}$ Create idle pickers set: $IP_t = \{1, \ldots, k\}$
Is there any order which has not been batched?
Yes
Define the batches set $B_t$ using batching rule ($H_B$), Calculate the service time for each batch by routing strategy ($St_j$)
$|B_t| > |IP_t|$?
Yes
No
Assign batches $B_t$ to the idle pickers using assigning rule $H_A^1$
$|B_t| < |IP_t|$?
Yes
No
Assign batches $B_t$ to the idle pickers using assigning rule $H_A^2$
$|B_t| = |IP_t|$?
Yes
$i = m$?
Yes
No
Assign batches $B_t$ to the idle pickers using assigning rule $H_A^3$
Update $B_t$, $I_t$ and $IP_t$
$i = m$ & $|B_t| = 0$?
Yes
No
All orders are processed
$|B_t| = |IP_t| = 1$?
Yes
No
Wait until the threshold time of the batch lapses or the next decision point event occurs earlier
Did the end of waiting threshold occur earlier?
Yes
No
Assign the remained batch using assigning rule $H_A^2$
Wait for the next decision point to occur
Update $t$, $B_t$, $I_t$ and $IP_t$

FIGURE 2. Flowchart of the on-line rule based algorithm

## 4.4. Assigning rules ($H_A$)

In this section we propose three assigning rules namely $H_A^1$, $\mathrm{H}_A^2$ and $H_A^3$ which will be used in the on-line order batching algorithm in order to assign the generated batches to the order pickers at each decision point. According to the status of the idle pickers and available unprocessed batches at each decision point, assigning rules decide whether each idle order picker should start processing one of unprocessed batches or wait for new customer orders to come. At each decision point the order picking system may have one of the following three states:

State 1: Number of unprocessed batches is greater than the number of idle pickers.
State 2: Number of unprocessed batches is less than the number of idle pickers or they are equal and the last order in known (all customer orders has been received).

State 3: Number of unprocessed batches is equal to the number of idle pickers and there is at least one customer order which is not known (at least one customer order has not been received yet).

According to the fact that state 1, state 2 or state 3 is encountered in a decision point, one or two rules among $H_A^1$, $H_A^2$ and $H_A^3$ are applied to assign the batches to the order pickers. Definition of three assigning rules are as follows:

$H_A^1$: Select a batch for each idle order picker by using selection rule and assign them to the order pickers to start the picking process of the selected batches immediately.

$H_A^2$: Assign all unprocessed batches to auxiliary idle order pickers to start the picking process of all batches immediately.

$H_A^3$: Define a waiting threshold time for each unprocessed batch according to the equality $\mathrm{WT}_j = 2r_i + \mathrm{st}_j - \mathrm{st}_i$. In this equation, $j$ represents the index of the batch and $i$ represents the index of an order in batch $j$ with the longest service time in comparison to the other orders in this batch. $r_i$ represents the arrival time of customer order $i$. $\mathrm{st}_j$ and $\mathrm{st}_i$ represent the service time of batch $j$ and order $i$, respectively. Select the batch with the maximum waiting threshold value and an auxiliary idle order picker to wait for probable upcoming orders according to the waiting threshold value of the batch. Then assign the remained batches to other idle order pickers to start the picking process of them immediately. If there is only one batch and one idle order picker, according to this rule, the idle order picker and the batch will be waiting for probable upcoming orders. While an idle order picker and a batch are waiting, if the waiting threshold time lapses before the occurrence of a new decision point, the idle order picker will immediately start to pick the batch.

Given that the order picking system is in the first, second or third state at a decision point, rules $H_A^1$, $H_A^2$ or $H_A^3$ will be applied, respectively. Flowchart of the proposed on-line algorithm consisting of order batching rule, batch selection rule and picker assigning rules is depicted in Figure 2. $I_t$, $B_t$ and $IP_t$ in Figure 2 represent the sets of unprocessed customer orders, unprocessed batches and idle order pickers at time epoch $t$, respectively.

## 5. Competitive analysis

### 5.1. Scope

The performance of on-line algorithms is commonly evaluated through competitive analysis approach [9]. According to this approach, on-line algorithm is compared to an algorithm which defines the optimal solution of the off-line version of the same problem with a complete sequence (optimal off-line algorithm). For a minimization problem, an on-line algorithm $F$ is called $c$-competitive if a constant $b$ exists such that for any instance $I$ the inequality $F(I) \leq c.\mathrm{OPT}(I) + b$ holds, where $F(I)$ represents the objective value obtained by on-line algorithm $F$ for instance $I$ and $\mathrm{OPT}(I)$ is the objective function value of an optimal solution for instance $I$ of the off-line problem. The competitive ratio of the on-line algorithm $F$ is the minimum value of $c$ for which the inequality holds.

(1) *Number of unprocessed batches is less than the number of idle order pickers*: In this case, according to the definition of decision points in Section 4.1, there is no unprocessed batch and there is at least one idle order picker in the order picking system. According to the on-line algorithm, the picking process of customer order $n$ is immediately started by one of the idle order pickers. If all of order pickers are idle (all of customer orders except customer order $n$ have been already completed) at $r_n$, the maximum completion time of all batches will be $F^*(I) = r_n + \mathrm{st}_n = \mathrm{OPT}(I)$, where $\mathrm{st}_n$ represents the time required to pick the last customer order. If there is at least one busy order picker at $r_n$, the state of the order picking system after the arrival of customer order $n$ is immediately turned to a special type of the second type of states which will be discussed in the following.

(2) *Number of unprocessed batches equals the number of idle pickers*: In this case, we know that according to the definition of decision points in Section 4.1, there would be one unprocessed batch or no unprocessed

batch. If there is no unprocessed batch at $r_n$, the state of the system will immediately turn to a special type of the third state type which will be discussed later. Otherwise, if customer order $n$ cannot form a single batch with customer orders in the present batch the state of the system immediately becomes a special type of the third state type, but if customer order $n$ can be added to the present unprocessed batch, it means that the last customer order arrives before the end of waiting threshold time of the batch, *i.e.* $r_n \leq 2r_{i_j} + \mathrm{st}_{i_j} - \mathrm{st}_j$ holds, Where $i_j$ is an order in this batch which has the longest service time. Moreover, the picking process of the last batch including the last customer order will be immediately started. The following equation holds for the maximum completion time of all batches:

$$\begin{aligned} F^*(I) &= \max\{r_n + \mathrm{st}_{j\cup n}, f_{\widetilde{1}}, f_{\widetilde{2}}, \ldots, f_{\widetilde{n-1}}\} \\ &\leq \max\{r_n + \mathrm{st}_j + \mathrm{st}_n, f_{\widetilde{1}}, f_{\widetilde{2}}, \ldots, f_{\widetilde{n-1}}\} \end{aligned}$$

where $f_{\widetilde{j}}$ represents the completion time of batch $\widetilde{j}$ which is being processed at $r_n$ for all $\widetilde{j} \in \{\widetilde{1}, \widetilde{2}, \ldots, \widetilde{n-1}\}$. Let $s_{\widetilde{j}}$ be the start time of batch $\widetilde{j}$ and let $P_{s_{\widetilde{j}}}$ and $\mathrm{st}^*_{P_{s_{\widetilde{j}}}}$ be the set of unprocessed customer orders at time $s_{\widetilde{j}}$ and the optimal total service time of these orders, respectively. It is clear that the following inequalities hold:

$$s_{\widetilde{j}} \leq r_n, \mathrm{st}_{\widetilde{j}} \leq \mathrm{st}^*_{P_{s_{\widetilde{j}}}}, \forall \widetilde{j} \in \{\widetilde{1}, \widetilde{2}, \ldots, \widetilde{n-1}\}, r_n \leq \mathrm{OPT}(I).$$

And also we know that the following inequalities hold:

$$\mathrm{st}^*_{P_{s_{\widetilde{j}}}} \leq \mathrm{OPT}(I), \forall \widetilde{j} \in \{\widetilde{1}, \widetilde{2}, \ldots, \widetilde{n-1}\}.$$

If $\max\{r_n + \mathrm{st}_j + \mathrm{st}_n, f_{\widetilde{1}}, f_{\widetilde{2}}, \ldots, f_{\widetilde{n-1}}\} = f_{\widetilde{j}'}$ holds, for some $\widetilde{j}' \in \{\widetilde{1}, \widetilde{2}, \ldots, \widetilde{n-1}\}$ the following is obtained:

$$F^*(I) \leq f_{\widetilde{j}'} = s_{\widetilde{j}'} + \mathrm{st}_{\widetilde{j}'} \leq r_n + \mathrm{st}^*_{P_{s_{\widetilde{j}'}}} \leq \mathrm{OPT}(I) + \mathrm{OPT}(I) = 2\mathrm{OPT}(I).$$

Otherwise if $\max\{r_n + \mathrm{st}_j + \mathrm{st}_n, f_{\widetilde{1}}, f_{\widetilde{2}}, \ldots, f_{\widetilde{n-1}}\} = r_n + \mathrm{st}_j + \mathrm{st}_n$ then one obtains:

$$F^*(I) \leq r_n + \mathrm{st}_j + \mathrm{st}_n \leq 2r_{i_j} + \mathrm{st}_{i_j} - \mathrm{st}_j + \mathrm{st}_j + \mathrm{st}_n = 2r_{i_j} + \mathrm{st}_{i_j} + \mathrm{st}_n.$$

If $\mathrm{st}_{i_j} \geq \mathrm{st}_n$ holds, the upper bound for $F^*(I)$ can be estimated as follows:

$$F^*(I) \leq 2r_{i_j} + \mathrm{st}_{i_j} + \mathrm{st}_n \leq 2r_{i_j} + 2\mathrm{st}_{i_j} \leq 2\mathrm{OPT}(I).$$

Otherwise the upper bound for $F^*(I)$ can be obtained as follows:

$$F^*(I) \leq 2r_{i_j} + \mathrm{st}_{i_j} + \mathrm{st}_n \leq 2r_{i_j} + 2\mathrm{st}_n \leq 2\mathrm{OPT}(I)$$

(3) *Number of unprocessed batches is larger than the number of idle pickers*: In this case, there are one or several unprocessed batches and no order pickers in the order picker system at $r_n$. Let $s_{\widetilde{j}}$ and $f_{\widetilde{j}}$ denote the start time and the completion time of batch $\widetilde{j} \in \{\widetilde{1}, \widetilde{2}, \ldots, \widetilde{n}\}$ which is being processed at time $r_n$ by the $j$th order picker and $I_{\widetilde{j}}$ be the set of customer orders assigned to batch $\widetilde{j}$. Let $P_{s_{\widetilde{j}}}$ and $\mathrm{st}^*_{P_{s_{\widetilde{j}}}}$ be the set of unprocessed customer orders at time $s_{\widetilde{j}}$ and the optimal total service time of these orders, respectively. Consider $\widetilde{j}'$ as the batch which has the minimum completion time among all the batches which are being processed at $r_n$. Define $r$ as the earliest arrival time of customer orders in the time interval $(s_{\widetilde{j}'}, f_{\widetilde{j}'}]$, *i.e.* $r = \min_i\{r_i | s_{\widetilde{j}'} \leq r_i \leq f_{\widetilde{j}'}\}$. Let $A(r)$ be the set of customer orders which arrive after $r$, more formally $A(r) = \{i | r_i \geq r\}$. The maximum completion time of all customer orders is obtained using the following equation:

$$F^*(I) = \max\left\{f_{\widetilde{j}'} + \mathrm{st}^*_{A(r)\cup P_{s_{\widetilde{j}'}} \setminus I_{\widetilde{j}}}, f_{\widetilde{1}}, \ldots, f_{\widetilde{j-1}}, f_{\widetilde{j+1}} \ldots, f_{\widetilde{n}}\right\}$$

where $\mathrm{st}^*_{A(r)\cup P_{s_{\widetilde{j'}}}\setminus I_{\widetilde{j}}}$ represents the optimal service time of all customer orders which arrive after the start time of batch $\widetilde{j'}$. In the above equation, if $F^*(I) = f_{\widetilde{j''}}$ for some $j'' \in \{\widetilde{1},\ldots,\widetilde{j-1},\widetilde{j+1},\ldots,\widetilde{n}\}$, the following is obtained:

$$F^*(I) = f_{\widetilde{j''}} = s_{\widetilde{j''}} + \mathrm{st}_{\widetilde{j''}} \leq r_n + \mathrm{st}^*_{P_{s_{\widetilde{j''}}}} \leq \mathrm{OPT}(I) + \mathrm{OPT}(I) = 2\mathrm{OPT}(I).$$

Otherwise, since $\mathrm{st}^*$ is defined by the optimal order batching algorithm so we have:

$$\begin{aligned} F^*(I) &= f_{\widetilde{j'}} + \mathrm{st}^*_{A(r)\cup P_{s_{\widetilde{j'}}}\setminus I_{\widetilde{j}}} \leq f_{\widetilde{j'}} + \mathrm{st}^*_{A(r)} + \mathrm{st}^*_{P_{s_{\widetilde{j'}}}\setminus I_{\widetilde{j}}} \\ &= s_{\widetilde{j'}} + \mathrm{st}^*_{I_{\widetilde{j'}}} + \mathrm{st}^*_{P_{s_{\widetilde{j'}}}\setminus I_{\widetilde{j}}} + \mathrm{st}^*_{A(r)} \\ &= s_{\widetilde{j'}} + \mathrm{st}^*_{P_{s_{\widetilde{j'}}}} + \mathrm{st}^*_{A(r)} \leq r + \mathrm{st}^*_{A(r)} + \mathrm{st}^*_{P_{s_{\widetilde{j'}}}}. \end{aligned}$$

Furthermore, the inequalities $r + \mathrm{st}^*_{A(r)} \leq \mathrm{OPT}(I)$ and $\mathrm{st}^*_{P_{s_{\widetilde{j'}}}} \leq \mathrm{OPT}(I)$ hold. Therefore one obtains:

$$F^*(I) \leq r + \mathrm{st}^*_{A(r)} + \mathrm{st}^*_{P_{s_{\widetilde{j'}}}} \leq \mathrm{OPT}(I) + \mathrm{OPT}(I) \leq 2\mathrm{OPT}(I).$$

To summarize, the competitive ratio of the proposed on-line algorithm is less than or equal to 2.

## 6. Numerical experiments

### 6.1. Purpose

An extensive series of numerical experiments has been conducted by considering different problem classes to simulate the behavior of the order picking system when using different batching, batch selection and picker routing methods. Since, according to our knowledge, there is no algorithm in the literature for on-line order batching problem with multiple pickers regarding makespan minimization and given that we have proved the competitiveness of the proposed on-line algorithm previously, we are suffice to compare performance of different batching, batch selection and routing methods using our proposed on-line algorithm through numerical experiments.

### 6.2. Experiment parameters

A single block warehouse with two cross aisles, one in the front and one in the back of the picking area, is considered which is frequently used in the literature. The picking area consists of 900 storage locations where a different storage location is considered for each article. These storage locations are arranged in 10 aisles in a way that 45 storage locations are included in each side of each aisle. The length of each storage location is one length unit (1 LU) and the center to center distance between two adjacent aisles is 5 LU. The depot is located 1.5 LU away from the first storage location of the leftmost aisle (aisle 1) and the distance between the front cross-aisle and the depot is 0.5 LU.

It is also assumed that there are two order pickers in the warehouse. Each order picker can walk 20 LU per unit of time and 10 s is needed for an order picker to collect an item from a storage location *i.e.* $v_{\text{travel}} = 48\,\mathrm{LU}\,\mathrm{min}^{-1}$ and $v_{\text{pick}} = 6$ items $\mathrm{min}^{-1}$. The setup time for each batch is 3 min.

A class-based storage assignment is considered to store items in storage locations. According to this policy, the articles are grouped into three classes A, B and C based on their demand frequencies. Articles with high demand frequency are included in class A and are stored in the first aisle. Articles with medium demand frequency are included in class B and are stored in the three subsequent aisles (aisles no.2, no.3 and no.4). Class C contains articles with low demand frequency which are stored in 6 remaining aisles. It is assumed that 52% of the requested articles belong to the articles in class A, 36% to the articles in class B and 12% to the articles in

Table 1. Parameter values and algorithms used for experiments.

| Parameters(Algorithms) | Values |
|---|---|
| Batch capacity $W$ (items) | 45, 75 |
| Total number of customer orders $n$ | 60, 120, 180, 240 |
| Routing algorithm | S-Shape, Largest Gap |
| Batching algorithm | FCFS, C&W(ii), ILS |
| Batch selection algorithm | FIRST, LONG, SHORT, SAV |

class C. It is also assumed that articles within a class are stored randomly. These assumptions are in line with those of Henn [9].

Several scenarios are defined using different values for problem parameters in order to analyze the quality of solutions obtained by the proposed on-line algorithm. Two different values are considered for batch capacity which are 45 and 75. The S-Shape and the Largest-Gap heuristic algorithms are used as the routing strategy. It is assumed that the number of items in each customer order follows a uniform distribution in $\{5,6,\ldots,25\}$. Considering the above defined capacities for the picking device and the aforementioned uniform distribution for the number of items in each customer order, there will be 3 or 5 customer orders in each batch averagely.

Four different values 60, 120, 180 and 240 are considered for the total number of customer orders. It is assumed that customer orders arrive within an 8-h shift. The interarrival times (the time between the arrival of two consecutive orders) follow the exponential distribution with parameter $\lambda$ as the arrival rate. Let $X(t)$ be the number of arriving orders in the time interval $[0,t]$. Considering the exponential distribution for interarrival times, $E[X(t)] = \lambda.t$ holds. To be in line with assumptions in [9], $\lambda$ is chosen in a way that the expected arriving orders for each order picker ($\frac{E[X(t)]}{k}$), is equal to $n$ for $t = 8\,\text{h}$. Therefore, the following values are used for $\lambda$: for $n = 60$: $\lambda = 0.0625$, for $n = 120$: $\lambda = 0.125$, for $n = 180$: $\lambda = 0.1875$ and for $n = 240$: $\lambda = 0.25$.

Eight problem classes are resulted with the combination of the test parameters which described above. For each problem class 10 instances have been generated which results in 80 different test instances in total. As summarized in Table 1, two batching algorithms, three batch selection methods and four picker routing heuristics are considered which results in 24 combinations of algorithms to be used for solving the problem. Each test instance is solved using each of 24 combinations of the algorithms. It means that the heuristic algorithm is implemented $80 \times 24 = 1920$ times.

All of these runs of the heuristic algorithm are carried out using Matlab R2014a on a Core i7 processor 1.6 GHz and 4.0 GB RAM

## 6.3. Experiment results

Considering all combination of the test instances which are defined in above and different algorithms which are used, the simulation program was run 1920 times. Run time of the program was 68 s in the worst case, justifying the need for a fast algorithm for the on-line problem. The average values of the maximum completion time of all batches (makespan) for all problem classes considering different algorithms are illustrated in Table 2. Problem classes and different algorithms have been shown in columns and rows of the table, respectively. In this table, For each combination of routing and batching algorithms, the best values which are obtained for makespan in each column (each problem class) have been distinguished with bold numbers. Also for each combination of routing and batch selection methods, the best values achieved for makespan in each column (each problem class) have been shown with underlined numbers. In the following, we will compare the performance of the algorithms in each group (the groups of batching algorithms, batch selection algorithms and routing algorithms). To compare different utilized algorithms in each group, two other groups of algorithms will be considered as constant.

Considering the bold numbers in Table 2, we can realize that regardless of the routing and the batching algorithms considered, the FIRST selection rule has obtained the equal or lower makespan as compared to three

Table 2. The average of maximum completion time (makespan) for all problem classes considering different algorithms.

| Routing | Batching | Selection | W=45, n= | | | | W=75, n= | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | 60 | 120 | 180 | 240 | 60 | 120 | 180 | 240 |
| S-Shape | FCFS | FIRST | **493** | **520** | **616** | **800** | **494** | **517** | **553** | **699** |
| | | SHORT | **493** | 530 | 665 | 856 | 496 | 530 | 607 | 757 |
| | | LONG | **493** | **520** | 620 | 802 | **494** | **517** | 556 | 702 |
| | | SAV | **493** | **520** | 621 | 801 | **494** | **517** | 557 | 703 |
| | C&W(ii) | FIRST | **493** | **516** | **572** | **706** | **494** | 514 | **530** | **617** |
| | | SHORT | 494 | 521 | 601 | 749 | 495 | 517 | 551 | 657 |
| | | LONG | **493** | 517 | 581 | 734 | **494** | 514 | 532 | 641 |
| | | SAV | **493** | 518 | 577 | 732 | **494** | **513** | 535 | 636 |
| | ILS | FIRST | 494 | 517 | 568 | 712 | **494** | 515 | **532** | 636 |
| | | SHORT | 494 | 517 | **556** | **682** | 495 | 522 | 540 | **622** |
| | | LONG | **493** | **516** | 584 | 739 | **494** | **513** | 536 | 646 |
| | | SAV | **493** | **516** | 583 | 740 | **494** | 514 | 533 | 643 |
| Largest Gap | FCFS | FIRST | **493** | **517** | **600** | **773** | **493** | **513** | **530** | **651** |
| | | SHORT | **493** | 527 | 653 | 834 | 494 | 524 | 579 | 726 |
| | | LONG | **493** | 518 | 602 | 776 | **493** | **513** | 532 | 656 |
| | | SAV | **493** | 518 | 602 | 778 | **493** | **513** | 532 | 655 |
| | C&W(ii) | FIRST | **493** | **515** | **567** | **703** | **493** | **511** | 527 | **613** |
| | | SHORT | 494 | 523 | 600 | 747 | 494 | 518 | 551 | 657 |
| | | LONG | **493** | **515** | 576 | 726 | **493** | 512 | **524** | 626 |
| | | SAV | **493** | 516 | 576 | 730 | **493** | **511** | 526 | 623 |
| | ILS | FIRST | 494 | **514** | **567** | 712 | **493** | **512** | **524** | **623** |
| | | SHORT | 494 | 520 | 573 | **678** | 494 | 517 | 539 | 631 |
| | | LONG | **493** | 515 | 574 | 728 | **493** | **512** | 525 | 629 |
| | | SAV | **493** | 516 | 576 | 731 | **493** | 513 | **524** | 628 |

other selection methods in 38 out of 48 cases (79.2% of the cases). If we want to look more closely, when using FCFS rule for batching customer orders, the FIRST selection rule has led to better results in all the cases (100% of the cases). When using C&W(ii) as the batching rule, in 14 out of 16 cases (87.5% of the cases) the FIRST selection has resulted in better makespan. Finally when using the ILS as the batching algorithm, in 8 out of 16 cases (50% of the cases classes) the FIRST selection rule has had better results. The LONG selection rule has achieved the equal or better results compared to three other selection methods in 16 out of 48 cases (33.3% of the cases). the SAV and the SHORT selection rules have occupied the third and the fourth rank with equal or better makespan to other methods in 12 cases out of 48 cases (25% of the cases) and 6 cases out of 48 cases (12.5% of the cases), respectively. An explanation for the superiority of the FIRST selection rule over other three selection methods can be that the order batching algorithms first try to improve the first constructed batch and then the subsequent batches.

In addition, according to the Table 2, it can be seen that regardless of the batching and the batch selection rules considered, the results obtained using the Largest Gap routing policy is at least as good as the results obtained using the S-Shape routing policy in almost all the cases (94.8 % of the cases). Figure 3 represent the superiority of the Largest Gap routing policy over the S-Shape routing policy considering different combinations of batching and batch selection methods for $W = 45$ and $W = 75$, respectively. The superiority of the Largest Gap routing policy over the S-Shape routing policy for each combination of batching methods, batch selection methods and the problem classes is calculated using the following formula $\mathrm{SV} = \mathrm{MS}_{\mathrm{SS}} - \mathrm{MS}_{\mathrm{LG}}$, Where SV, $\mathrm{MS}_{\mathrm{LG}}$ and $\mathrm{MS}_{\mathrm{SS}}$ represent the superiority value, the makespans which have been obtained using the Largest

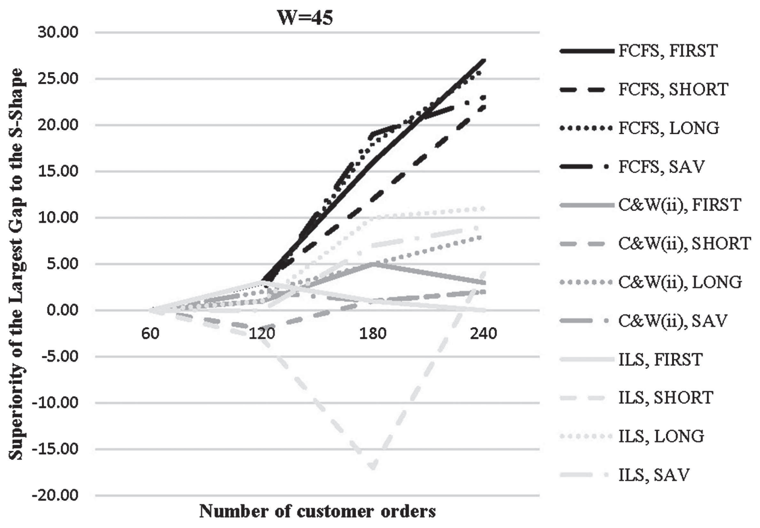

Figure 3. Superiority of the Largest Gap routing policy to the S-Shape routing policy for $W = 45$.

Gap and the S-Shape routing policies, respectively (all in minutes). As it can be seen in Figure 3, except three cases (using ILS and SHORT methods with $n = 120$, using ILS and SHORT methods with $n = 180$ and using C&W(ii) and SHORT methods with $n = 120$) in all other cases performance of the Largest Gap algorithm is equal or better than the S-Shape algorithm. Also, as it can be seen in Figure 4, except two cases (using C&W(ii) and SHORT methods with $n = 120$ and using ILS and SHORT methods with $n = 240$) in all other cases the Largest Gap routing policy has equal or better performance than the S-Shape routing policy. Moreover, considering both Figures 3 and 4 one can realize that in problem classes with smaller number of customer orders, the routing algorithms have nearly equal performance but as the number of customer orders increases, the Largest Gap routing policy shows great superiority over the S-Shape routing policy up to nearly 25 min and 50 min for $W = 45$ and $W = 75$, respectively.

Considering the underlined numbers in Table 2, if we compare the batching methods regardless of the routing and the selection strategies which are considered, in 15 out of 64 cases (23.4% of the cases), the FCFS batching rule have reached the equal or lower makespan as compared to two other methods which all of them are related to the problem classes with the smallest number of customer orders ($n = 60$). The C&W(ii) algorithm have had equal or better results in comparison to two other methods in 49 out of 64 cases (76.5% of the cases). The ILS algorithm also have achieved the equal or lower makespan as compared to two other methods in 37 out of 64 cases (57.8% of the cases). To summarize, the C&W(ii) and the ILS algorithms have shown great superiority over the FCFS rule in batching customer orders specially in the problem classes with more than 60 customer orders.

## 7. Conclusion and future scope

This paper considers the on-line order batching and scheduling problem with multiple pickers in a manual order picking warehouse. The challenge is to form batches from dynamically arriving orders, assign them to the order pickers and schedule the picking process of the batches assigned to each picker in a way that the maximum completion time of all batches (makespan) is minimized. Considering the maximum completion time of all batches (makespan) as the objective function causes the presented model to be of practical importance because from one point of view it can reduce the regular and overtime working hours of the order pickers and from another point of view it can reduce the delivery lead times which results in increased service level. First,

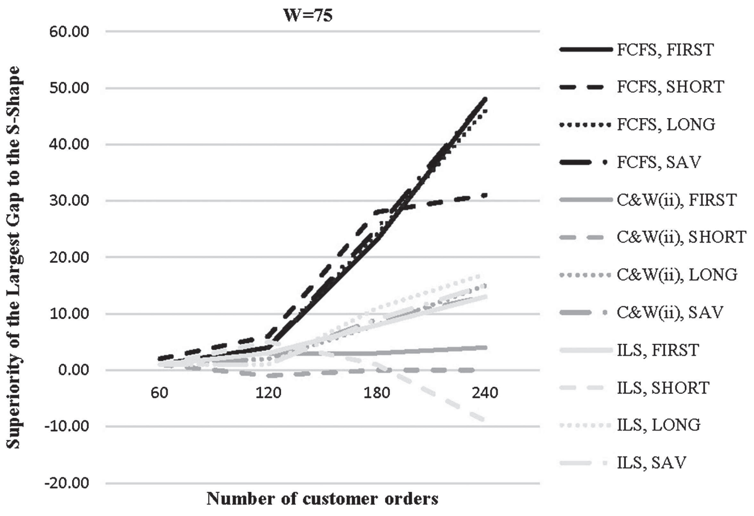

FIGURE 4. Superiority of the Largest Gap routing policy to the S-Shape routing policy for $W = 75$.

a mathematical model is presented to represent the off-line version of the problem. Then a novel rule-based on-line algorithm is proposed to solve the on-line problem which is composed of a batching algorithm, a batch selection rule, a picker routing policy and several rules for picker assigning. By means of a competitive analysis it is shown that the competitive rate of the proposed on-line algorithm equals 2. Since this is the first work which tries to minimze the maximum completion time of all customer orders considering multiple pickers and there is no similar work in the literature, Through an extensive numerical experiments it has been evaluated which batching algorithm, which batch selection rule and which routing policy lead to best result.

A future research object should be to extend the presented model in this paper to consider other objective functions beside makespan and to propose appropriate algorithms to solve the multi-objective problem. The second direction for the future research could be to integrate the order batching and vehicle routing problems and optimize them jointly.

## REFERENCES

[1] Y.A. Bozer and J.W. Kile, Order batching in walk-and-pick order picking systems. *Int. J. Prod. Res.* **46** (2008) 1887–1909.

[2] F. Caron, G. Marchet and A. Perego, Routing policies and coi-based storage policies in picker-to-part systems. *Int. J. Prod. Res.* **36** (1998) 713–732.

[3] E.P. Chew and L.C. Tang, Travel time analysis for general item location assignment in a rectangular warehouse. *Eur. J. Oper. Res.* **112** (1999) 582–597.

[4] M.B.M. De Koster, E.S. Van der Poort and M. Wolters, Efficient orderbatching methods in warehouses. *Int. J. Prod. Res.* **37** (1999) 1479–1504.

[5] E.A. Elsayed and M.-K. Lee, Order processing in automated storage/retrieval systems with due dates. *IIE Trans.* **28** (1996) 567–577.

[6] E.A. Elsayed, Algorithms for optimal material handling in automatic warehousing systems. *Int. J. Prod. Res.* **19** (1981) 525–535.

[7] N. Gademann and S. Velde, Order batching to minimize total travel time in a parallel-aisle warehouse. *IIE Trans.* **37** (2005) 63–75.

[8] D.R. Gibson and G.P. Sharp, Order batching procedures. *Eur. J. Oper. Res.* **58** (1992) 57–67.

[9] S. Henn, Algorithms for on-line order batching in an order picking warehouse. *Comput. Oper. Res.* **39** (2012) 2549–2563.

[10] S. Henn and V. Schmid, Metaheuristics for order batching and sequencing in manual order picking systems. *Comput. Ind. Eng.* **66** (2013) 338–351.

[11] S. Henn, S. Koch, K.F. Doerner, C. Strauss and G. Wäscher, Metaheuristics for the order batching problem in manual order picking systems. *Bus. Res.* **3** (2010) 82–105.

[12] C.-M. Hsu, K.-Y. Chen and M.-C. Chen, Batching orders in warehouses by minimizing travel distance with genetic algorithms. *Comput. Ind.* **56** (2005) 169–178.

[13] N. Kamin, On-line Optimization of Order Picking in an Automated Warehouse. Shaker, Herzogenrath (1998).

[14] T. Le-Duc and R.M. de Koster, Travel time estimation and order batching in a 2-block warehouse. *Eur. J. Oper. Res.* **176** (2007) 374–388.

[15] C.-Y. Tsai, J.J.H. Liou and T.-M. Huang, Using a multiple-ga method to solve the batch picking problem: considering travel distance and order due time. *Int. J. Prod. Res.* **46** (2008): 6533–6555.

[16] I. Van Nieuwenhuyse and R.B.M. de Koster, Evaluating order throughput time in 2-block warehouses with time window batching. *Int. J. Prod. Res. Econ.* **121** (2009) 654–664.

[17] G. Wäscher, Order picking: a survey of planning problems and methods. In: Supply Chain Management and Reverse Logistics. Springer, Berlin (2004) 323–347.

[18] J. Won and S. Olafsson, Joint order batching and order picking in warehouse operations. *Int. J. Prod. Res.* **43** (2005) 1427–1442.

[19] X. Xu, T. Liu, K. Li and W. Dong, Evaluating order throughput time with variable time window batching. *Int. J. Prod. Res.* **52** (2014) 2232–2242.

[20] M. Yu and R.B.M. De Koster, The impact of order batching and picking area zoning on order picking system performance. *Eur. J. Oper. Res.* **198** (2009) 480–490.

[21] J. Zhang, X. Wang and K. Huang, Integrated on-line scheduling of order batching and delivery under b2c e-commerce. *Comput. Ind. Eng.* **94** (2016) 280–289.